ALLOGÈNE : UN ENVIRONNEMENT D'APPRENTISSAGE DE L'ALGORITHMIQUE.

Jean-Pierre FOURNIER, Jacky WIRZ

Résumé : ALLOGÈNE est un environnement informatisé dédié à l'apprentissage de l'algorithmique pour les débutants. Au-delà d'une simple aide à la programmation, une assistance est réalisée dès l'accès à l'énoncé du problème algorithmique. Un hyperénoncé que l'on doit reformuler avant de passer à la modélisation qui se pratique avec un éditeur graphique permettant l'édition et l'interprétation de tout ou partie des instructions ainsi préparées. ALLOGÈNE est décomposé en modules EAO basés sur la démarche algorithmique proposée. Un cadre de travail convivial, un appui constant de la machine permettent ainsi de former un débutant aux concepts de base qu'il devra mettre en oeuvre dans les environnements professionnels qu'il sera amené à utiliser.

LA DÉMARCHE ALLOGÈNE

Nous définissons l'apprentissage de l'algorithmique comme l'acquisition des savoir-faire liés à la description des actions à effectuer pour atteindre un but dans un univers de ressources limitées imposant des contraintes d'utilisation en rapport avec les actions désirées. Partant de l'énoncé d'un problème, notre enseignement de l'algorithmique se propose de fournir les outils et méthodes permettant de trouver une solution applicable dans un monde spécifique. Par étapes successives, l'apprenant sera amené à passer de l'énoncé du problème, formulé en langue naturelle, à la modélisation de sa solution exprimée d'abord dans un langage de description puis avec un langage de modélisation directement assimilable par la machine. L'algorithme ainsi établi sera pris en charge par un automate programmable pour sa validation. Prenant pour base cette stratégie, l'élaboration d'un algorithme dans le cadre d'une introduction à l'algorithmique avec ALLOGÈNE se décompose donc en quatre étapes successives :

1- Consulter : lecture et compréhension de l'énoncé du **problème algorithmique**[1].

2- Reformuler : analyse du problème avec **reformulation** orientée machine.

3- Modéliser : **modélisation d'une solution** avec l'édition de l'algorithme.

4- Valider : interprétation de l'algorithme par un automate pour validation.

---

[1]. Les termes en gras dans le texte font l'objet d'une note explicative en fin de document.

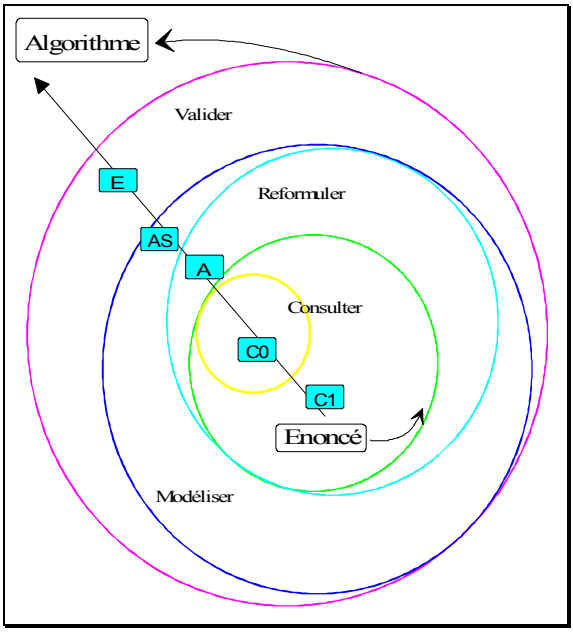
Ces quatre étapes impliquent, pour l'apprenant, de traverser les différents niveaux séparant les domaines cognitifs mis en oeuvre lors de l'élaboration de son algorithme, ce que nous pouvons présenter sous la forme du tableau ci-après. L'étape "Publier" met fin à "Valider" et signifie que l'apprenant prend la responsabilité du bon fonctionnement de son algorithme.

ENONCE

ETAPES	**DOMAINES COGNITIFS**		**PRODUCTION**
	à pratiquer	à intégrer	de
Consulter	connaissance	compréhension	-
Reformuler	compréhension	application	énoncé reformulé
Modéliser	application	analyse-synthèse	modèle
Valider	analyse-synthèse	évaluations	sous-algorithmes
Publier	évaluation	-	

ALGORITHME

Il y a plusieurs façons de parcourir ce tableau. La manière la plus classique consiste à le "consommer" ligne par ligne. Bien qu'ALLOGÈNE puisse être pratiqué de cette manière, un tel parcours ne convient pas à une méthode essai-approfondissement, c'est-à-dire de raffinement, puisque le retour en arrière consiste à ignorer les étapes adjacentes. Revenir sur ses pas est bien souvent nécessaire pour : corriger un modèle partiel, tenir compte de nouvelles données, compléter pour maîtriser ses choix et voir même changer de stratégie pour avoir un traitement plus efficace, plus rigoureux. Ainsi, consulter, reformuler, modéliser et valider peuvent être des étapes lacunaires avant de "Publier". Ces lacunes sont soit volontaires, soit involontaires. Un cas de lacune volontaire est typiquement mis en évidence lors d'une résolution fragmentaire d'un problème (composant ou sous-problème dégradé [GRAM-86]). Se trouver dans une telle situation ne porte pas à conséquence, l'apprenant peut même y avoir été amené par une suggestion faite par l'énoncé. Quand à la "lacune involontaire", elle résulte avant tout d'une analyse incomplète ou imparfaite et est en rapport avec le domaine cognitif d'analyse-synthèse.



Spirale cognitive et domaines associés

C0 Connaissance

C1 Compréhension

A Application

AS Analyse et Synthèse

E Evaluation

Activités d'un apprenant utilisant ALLOGÈNE. Les points de tangence correspondent aux passages inter modules.

La réflexion portée sur les domaines cognitifs et la modélisation des niveaux associés a conduit à réunir ceux-ci selon un schéma géocentrique : la spirale cognitive (voir figure spirale cognitive). Chaque domaine cognitif est constitué d'un disque. Un disque, donc un domaine, possède un point de tangence avec le domaine qui le contient. L'apprenant s'active dans la surface du domaine pour y exploiter les ressources proposées par ALLOGÈNE et accède à un autre domaine en empruntant le point de tangence qui se trouve sur le cercle délimiteur. Il y a un sens de parcours privilégié, néanmoins, tant l'approche descendante qu'ascendante étant appliquée pour la résolution des problèmes algorithmiques les retours arrières sont toujours possibles. Ceux-ci permettent à l'apprenant de retrouver la connaissance antérieure dont un domaine était générateur puisque à chaque transition les acquis sont conservés par ALLOGÈNE.

UN DÉCOUPAGE MODULAIRE

Nous avons défini des modules EAO calqués sur la démarche d'apprentissage venant d'être décrite, permettant ainsi d'intégrer l'ordinateur lui-même dans le processus d'élaboration suivi par l'apprenant. L'environnement d'apprentissage ALLOGÈNE comporte donc :

1- Un module de présentation des énoncés pour le domaine consulter.

2- Un module d'aide à l'analyse pour le domaine de la reformuler.

3- Un module d'édition des algorithmes pour le domaine de la modéliser.

4- Un module d'interprétation pour le domaine de la valider.

Nous introduisons de plus :

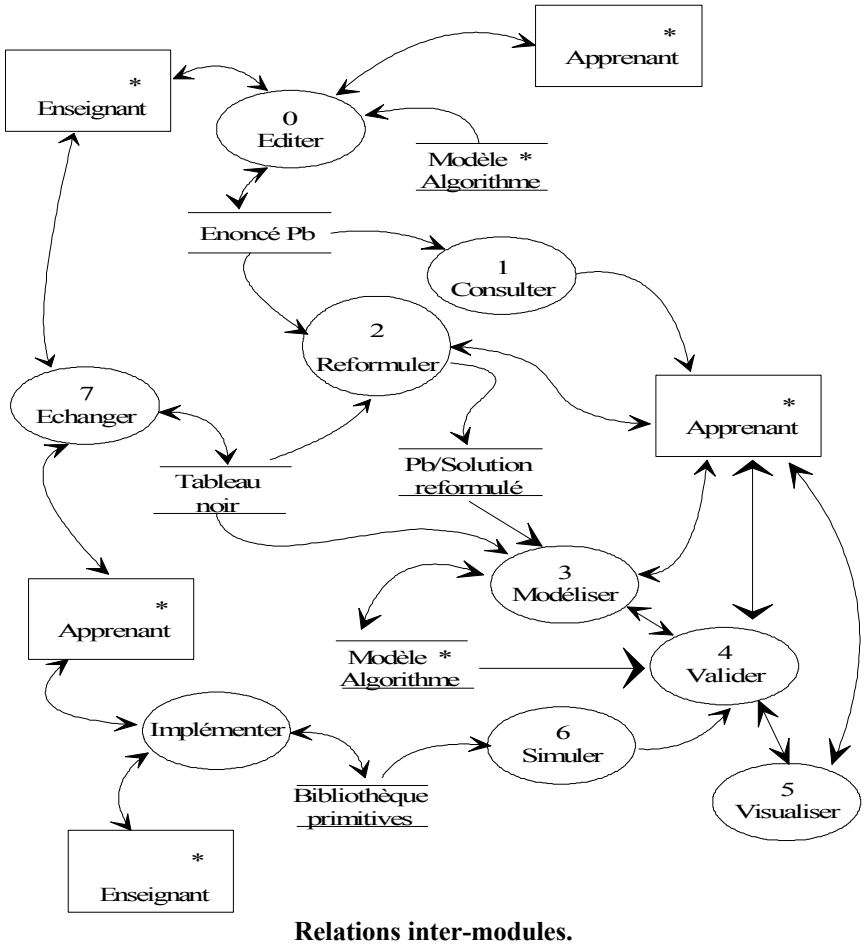
5- Un module de visualisation des objets manipulés par l'algorithme

6- Un module effectuant la simulation de divers processus, permettant ainsi la mise en oeuvre d'algorithmes d'un niveau de complexité satisfaisant ayant une éventuelle connotation ludique en regard de la pédagogie souhaitée. L'unité de simulation est appelée **machine virtuelle**.

Enfin, pour mettre en place une pédagogie "active" et de collaboration induisant une dynamique chez les apprenants, nous avons également adjoind :

7- Un module de communication entre apprenants, afin qu'ils puissent échanger des informations, sous-algorithmes, idées...

L'ensemble de ces sept modules forme l'environnement d'apprentissage de l'algorithmique, dont les relations intermodules sont décrites par la figure suivante.



Relations inter-modules.

Qui dit apprentissage dit évolution des savoir-faire. Nous avons donc prévu que cet environnement EAO soit évolutif pour qu'il puisse être adapté aux aptitudes réelles des apprenants. En d'autres termes cela signifie qu'il est totalement **paramètrable par l'enseignant** tant au niveau individuel (couple <poste apprenant>) qu'au niveau groupe (classe ou sous-groupes). Ce paramètrage agit entre autres sur :

- la présentation effectuée par le module de consultation des énoncés ;
- le niveau d'aide du module d'analyse ;
- la complexité du langage de modélisation des algorithmes (conceptuelle & syntaxique).

Trois niveaux sont proposés : "très débutant", débutant et "débutant avancé".

Nous avons prévu que certains modules ALLOGÈNE soient ouverts à la spécificité des enseignants. L'intervention de l'enseignant pourra se situer, bien entendu, au niveau de la rédaction des énoncés et de l'élaboration des modèles solution, mais également à celui des primitives offertes par le système au travers de la simulation de **machines virtuelles**.

LES MODULES EAO

Nous allons maintenant décrire chacune des entités EAO en termes d'objectif(s) à atteindre ainsi que d'une brève description. A titre d'illustration, nous utiliserons un exemple classique, retenu d'une part pour sa richesse exploratoire et, d'autre part, pour les extensions -non développées dans ce cadre- qu'il permet d'exploiter. Notre fil conducteur sera le problème algorithmique suivant :

Calculer le nombre de jours écoulés depuis le jour de nouvel-an en connaissant une date de cette année par un quantième, un mois et l'année.

Ce problème est destiné à un "débutant avancé" ; il fait appel en effet à des connaissances -structure alternative, initialisation d'une suite, etc.- préalablement acquises au travers de problèmes plus simples.

❑ MODULE 1, CONSULTER :

Objectif : Présentation à l'apprenant de l'énoncé du problème à traiter ainsi que conseils et méthodes. Ce module donne accès aux notions de bases du cours d'algorithmique au travers de livres électroniques.

Description : La présentation des énoncés est basée sur les concepts liés à l'hypertexte et hypergraphique. Les "hyperénoncés" définissent l'univers du problème ainsi que sa solution type.

Nombre de jours écoulés au cours d'une année

Présentation résumée :

Connaître le nombre de jours écoulés à une date donnée est d'une grande utilité dans plusieurs types d'applications. Le calcul d'une durée de travail -attribution de vacances- en est un exemple.Le calendrier est utilisé depuis fort longtemps pour identifier des périodes, fixer des rendez-vous, des échéances, etc. Les calendriers ont été établis par des personnes autorisées sur la base des mouvements des astres. Des réajustements de calendrier sont nécessaires périodiquement[2] ou exceptionnellement, en 1732 par exemple.

Enoncé :

Ecrire un algorithme fiable de calcul du nombre de jours écoulés pour une date complètement spécifiée.

Primitives autorisées :

*Opérateurs : + - * modulo < = ;*
Objets : nombre suite ;
Instructions : affectation alternative

Les énoncés doivent aussi permettre, par l'intermédiaire de leurs mots sensibles, d'expérimenter, de concrétiser la notion au centre du thème de l'algorithme à développer ; dans notre exemple, calendrier offre la possibilité de faire établir un calendrier pour une année choisie.

Ce module est ouvert à l'enseignant qui pourra ajouter ses propres sujets à ceux préexistants.

❑ **MODULE 2, REFORMULER :**

Objectif : Guider l'apprenant dans sa démarche d'analyse par identification et reformulation.

Description : Mise en oeuvre d'une technique d'analyse constituée des étapes suivantes : identifier les objets de l'univers du problème, déterminer les variants et invariants et trouver les transformations permettant de passer des données aux résultats, avec établissement de la liste des processus associés.

---

2. Dans cette présentation, le soulignement signifie que le mot est sensible à une action d'interaction -souris en mouvement, bouton souris pressé, touche clavier- ; par exemple, *périodiquement* mettra en évidence les années bissextiles et séculaires.

Suggestions :
1) résolvons le problème dégradé [3]pour des années toutes semblables et toutes constituées de 12 mois de 30 jours (la solution se résume à une simple expression)

2) considérons toutes les années ni bissextiles ni séculaires mais avec une distribution des jours du mois sur un intervalle de 28 à 31 jours ; pour un mois, nous pouvons appliquer une correction dépendante du numéro du mois :
- pour février, le nombre de jours écoulés est 30 + quantième + 1 ;
- pour mars ; le nombre de jours écoulés est 60 + quantième - 1 ;
- pour avril ; le nombre de jours écoulés est 90 + quantième ; etc.
Nous observons que l'expression de la suggestion 1) peut être réutilisée à condition de lui apporter une correction :
- quantième de février, le nombre de jours écoulés devient : jours écoulés + correction de février ;
- quantième de mars ; le nombre de jours écoulés devient : jours écoulés + correction de mars ;
- quantième d'avril ; le nombre de jours écoulés devient : jours écoulés + correction de avril ; etc.
Cette dernière façon de voir suggère la création d'une suite accédée par le numéro du mois de la date et initialisée des corrections correspondantes (somme des compléments à 30 de chaque mois).

3)prenons en compte les années bissextiles et corrigeons

4)prenons en compte les années séculaires et corrigeons

Ce texte est figé ; c'est le premier des deux composants de ce module dont le deuxième est un éditeur de reformulation qui peut être complété par un apprenant de cette façon :

Proposition 1
Avec 12 mois de trente jours, j'ai :
*les jours écoulés <- 30 * (numéro du mois - 1) + quantième - 1*

Proposition 2
La suite des corrections liées au mois est : (0, 1, -1, 0, 0, 1, 1, 2, 3, 3, 4, 4).
J'accède à une correction avec un indice de numéro de mois ; alors :
jours écoulés <- jours écoulés + correction du mois pour le numéro du mois

Proposition 3*.....*

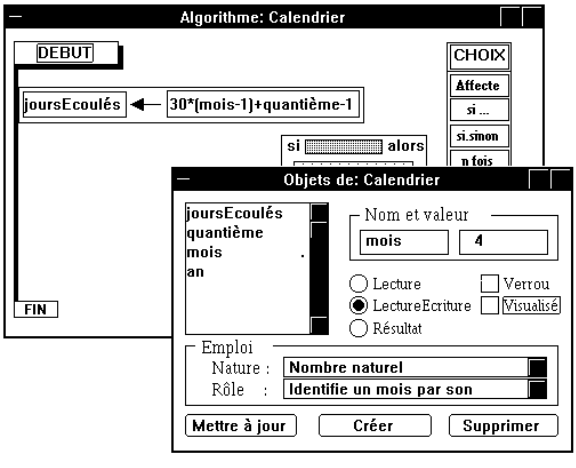
Ce module est considéré comme fermé, il est guidé par l'énoncé et son contenu a été défini lors de l'établissement de ce dernier.

3. Méthodologie basée sur les propositions du groupe Anna GRAM [GRA-86].

❑ **MODULE 3, MODÉLISER :**

Objectif : Permettre à l'apprenant de modéliser, par édition, sa solution afin de la soumettre à l'automate programmable pour validation.

Description : Mise en oeuvre d'un éditeur dédié, évolutif et guidé par la syntaxe dans sa version texte. Cet éditeur est en relation avec le module 4 afin de permettre une validation d'un sous algorithme La mise en oeuvre d'une édition graphique[4] des algorithmes est réalisée pour les niveaux "très débutant" et débutant.



Cet exemple montre l'avancement de la modélisation (proposition 1) ; il faut mettre en évidence que les objets -variables- sont déclarés dans une fenêtre dédiée à cet usage. La fenêtre masquée est la métaphore d'une table de montage ; le bord gauche met en évidence un "attracteur" d'instructions qui peuvent être interprétées (valider) par une commande ad hoc.

Ce module est fermé (il est constitué de l'éditeur graphique, commande l'interpréteur, etc.).

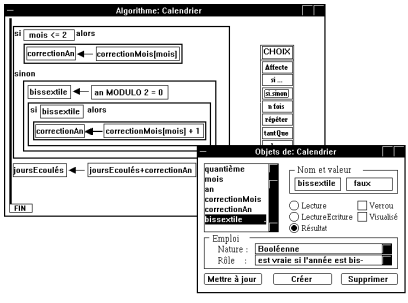
❑ **MODULE 4, VALIDER :**

Objectif : Assurer par des tests l'exactitude de l'algorithme dans des cas de figure préétablis.

Description : Mise en oeuvre un interpréteur dit évolutif. Cet interpréteur est classique en ce sens qu'il "exécute" les modèles construit à l'aide du module 3. Mais il peut également interpréter des modèles partiels de telle sorte que les apprenants puissent demander à la machine une aide en terme de "essayer pour voir". Sa particularité évolutive est le fait qu'il offre une possibilité d'extension graduelle de la syntaxe du langage de modélisation donc une évolution de la complexité des concepts (structure des instructions, des données) qu'il sera à même d'interpréter. Nous avons défini des niveaux de fonctionnement du type "très débutant", débutant, "débutant avancé" auxquels nous faisons correspondre une grammaire spécifique du

4. L'édition graphique des algorithmes ne correspond pas, pour nous, au dessin d'algorigrammes, mais à la manipulation d'objets (les primitives).

langage de modélisation. Bien entendu ces niveaux peuvent être sélectionnés par l'enseignant en terme d'étudiant ou groupe d'étudiants.

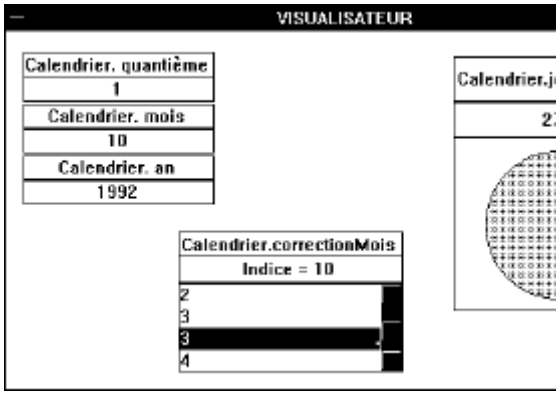


L'apprenant utilise l'imbrication des alternatives ; il peut tester une instruction ou un groupe d'instructions en effectuant une sélection. L'interpréteur exécute soit la séquence complète liée à l'attracteur, soit une sélection. De plus, des points d'arrêts peuvent être déposés pour une mise au point localisée, c'est-à-dire relative à la préoccupation du moment.

❑ **MODULE 5, VISUALISER :**

Objectif : Montrer le comportement interne du modèle solution au travers de l'évolution du contenu des objets manipulés par l'algorithme.

Description : Mise en oeuvre d'une représentation logique et/ou analogique des valeurs numériques manipulées par l'algorithme au fur et à mesure de son interprétation. Des barres, surfaces proportionnelles ainsi que des associations de couleurs sont utilisées.



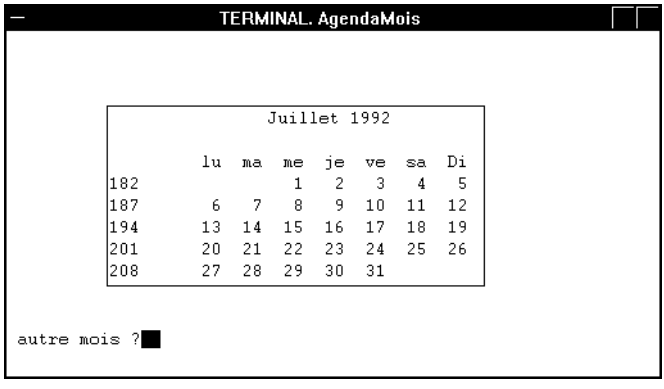
L'apprenant a utilisé le visualisateur pour disposer de plus d'informations sur ses objets : des représentations digitales -dont une de suite- et une représentation analogique -un diagramme sectoriel, qui affiche relativement le jours écoulés- ont été créées.

❑ **MODULE 6, SIMULER :**

Objectif : Fournir à l'apprenant un ensemble de primitives lui permettant de mettre en oeuvre ses algorithmes. Ces primitives sont regroupées logiquement par des machines virtuelles.

Description : Bibliothèques de primitives simulant des processus tels que : une table traçante, un terminal, machine-outil programmable, robot, instruments de mesure, comportement d'une automobile, train convoyeur, ascenseur, etc...

Ce module sera en relation avec le module 3 d'interprétation.



Un étudiant utilise la machine virtuelle 'TERMINAL' ; il a développé un algorithme 'AgendaMois' qui lui permet d'obtenir une feuille d'un mois de calendrier.

Ce module est ouvert aux enseignants pour qu'ils puissent y ajouter les primitives de leur choix en relation avec les énoncés des problèmes proposés.

❑ **MODULE 7, ECHANGER :**

Objectif : Fournir aux apprenants la possibilité d'échanger des matériels algorithmiques ainsi que des idées.

Description : Messagerie de type "tableau noir" avec en plus identification de l'auteur.

CONCLUSION

Notre travail de recherche portant sur la mise en place de nouvelles stratégies d'apprentissage pour un cours d'introduction à l'algorithmique, nous l'avons centré sur l'utilisation d'un environnement informatisé introduisant l'ordinateur comme un outil d'aide et non plus comme objet de l'enseignement par lui même, avec :

- l'introduction de l'ordinateur dès la prise en charge de l'énoncé (basé sur les techniques d'hypertexte) ;
- la mise en place d'une conception assistée d'algorithmes que nous avons appelé une aide à la reformulation ;
- la mise à disposition de la métaphore d'une table de montage des algorithmes ;
- l'intégration d'un outil de visualisation dynamique des objets manipulés par l'algorithme et une ou plusieurs machines virtuelles ;
- la mise en place d'une messagerie spécifique à l'échange d'algorithmes.

Notre approche se retrouve en partie dans les projets DIDALP[5], ou encore, ARCADE [PEY-88] pour l'aspect visualisation. La mise en oeuvre de la reformulation telle que proposée par le groupe Anna GRAM [GRAM-86] et supportée par des schémas d'induction [BEN-91] ainsi que l'utilisation de la métaphore d'une table de montage pour la modélisation des algorithmes en font l'originalité. De plus, l'homogénéité de l'environnement d'apprentissage depuis l'énoncé du problème algorithmique avec des références au cours, jusqu'à la validation du modèle solution par l'apprenant ainsi que par le professeur est le point fort d'ALLOGÈNE.

D'autre part, un des potentiels d'ALLOGÈNE réside en son ouverture sur des classes de problèmes variés pouvant être regroupés dans une base de données. Ajoutons encore que si le logiciel de base est prévu pour une didactique de l'algorithmique, il est néanmoins possible de l'utiliser comme un illustrateur de concepts plus avancés - par exemple : les processeurs, les transmissions de données dans une école d'ingénieurs - toujours sur une base d'hyperénoncés mais avec l'abandon des contraintes de reformulation ; l'aspect illustratif étant lié aux machines virtuelles et à la visualisation dynamique des objets.

Notes : Suit une brève description des renforcements *in text*

problème algorithmique

Pour l'environnement d'apprentissage ALLOGÈNE, les problèmes algorithmiques sont des énoncés de questions à propos desquels les apprenants devront trouver la méthode que devra suivre l'automate, cette méthode n'étant rien d'autre que l'algorithme lui même. Il est à noter qu'ALLOGÈNE impose, tout du moins suggère très fortement, une démarche analytique caractéristique pour résoudre les problèmes algorithmiques.

monde clos et avec contraintes

Lors de la rédaction d'un énoncé d'un problème algorithmique, l'auteur, le professeur, indique dans les attributs du problème les primitives autorisées. La contrainte, elle, est apportée par l'informatisation de l'environnement d'apprentissage lui-même qui permet d'utiliser telle primitive ou telle autre que si dans les attributs du problème algorithmique cette autorisation a été donnée par l'auteur. Il en est de même pour le langage de modélisation.

solution

Une solution viable est une suite finie de décisions et d'actes pouvant résoudre une difficulté : le problème algorithmique.

modélisation

L'assemblage et l'ordonnancement de la suite finie des éléments d'une solution définissent cette activité ; du point de vue "mécanique" celle-ci se pratique à l'aide de la métaphore d'une table de montage.

modèles solution

---

5. Le projet DIDALP a étét défini en 1985 par l'équipe dirigée par le professeur PC Scholl de l'université Joseph Fourier de Grenoble.

La juxtaposition des deux termes modèle et solution signifie que le modèle, l'algorithme modélisé, est solution du problème algorithmique. Dans ce contexte solution agit comme qualificatif substantivé à modèle.

reformulation

C'est l'activité qui consiste à trouver les éléments de la suite finie composant la solution.

paramétrable par l'enseignant

Le monde clos avec contraintes peut être défini par l'enseignant pour tout énoncé de problème (remarquons que c'est le cas aussi pour tout compilateur qui offre un nombre limité de ressources primitives).

machine virtuelle

Il en existe de deux types. Pour le premier, il s'agit d'un composant de type périphérique informatique -traceur,terminal- et pour le second ce sont davantage des réalisations simulées assimilables à des processus de contrôle pilotés par une machine informatique.

Jean-Pierre FOURNIER, Jacky WIRZ

Ecole d'Ingénieurs de Genève
4 rue de la praire, CH-1202 GENEVE.
Tél CH-22 344 77 50
Fax CH-22 344 92 88

eMail : **FOURNIERJP@eig.unige.ch**
eMail : **WIRZ@eig.unige.ch**

BIBLIOGRAPHIE

Note : les références en minuscules concernent des actes de conférence.

[ARS-88] *Préceptes pour programmer,* J. ARSAC ; Dunod - 1988

[BIO-81] *Introduction à la programmation* - Tome 1 « Algorithme et Langages », J. BIONDI & G. CLAVEL ; Masson - 1981

[BIO-84] *Introduction à la programmation* - Tome 2 « Structures de données », J. BIONDI & G. CLAVEL ; Masson - 1984

[BEN-91] *Systèmes formels Introduction à la logique et à la théorie des langages,* Cl. BENZAKEN ; Masson - Logique mathématiques informatique - 1991

[BRO-87] *Algorithm animation,* Marc H. BROWN ; The MIT Press - 1987

[cali-91] CALISCE 91 - *Proceedings International Conference on Computer Aided Learning and Instruction in Science and Engineering,* Presses polytechniques et universitaires romandes - 1991

[cfdi-90] *Deuxième colloque francophone sur la didactique de l'informatique,* Namur- 1990

[CHA-89] *Algorithmique - Cours et exercices méthodologiques corrigés*, GUY CHATY ET JEAN VICARD ; Collection informatique - NATHAN Université.

[eaor-89] *Enseignement et apprentissage avec l'ordinateur,* Fondazione Dalle Molle - Martigny / 1989

[DRE-84] *Intelligence artificielle : mythes et limites* H. L. DREYFUS ; Flammarion - 1984

[FEL-87] *La scolarisation de l'informatique à Genève,* DOMINIQUE FELDER ; Cahier N° 22 - 1987 ; Service de la recherche sociologique - Genève

[FEL-89] *L'informythique,* Dominique Felder ; Cahier N° 29 - 1989 Service de la recherche sociologique - Genève

[fiao-91] « Formation intelligemment assistée par ordinateur », *13èmes journées francophones sur l'informatique* - Genève/1991

[GRA-86] *Raisonner pour programmer groupe,* Anna GRAM ; Dunod informatique - 1986

[GOL-82] *Computer Science : A Modern Introduction,* Les Goldschlager + Andrew LISTER ; Prentice/Hall - 1982 (en français : *Informatique et Algorithmique* ; InterEditions - 1986)

[MIN-88] *La société de l'esprit,* Marvin MINSKY ; InterÉditions - 1988 (*The Society of Mind* ; Simon & Schuster - New York/1985)

[PAP-81] *Jaillissement de l'esprit*, *Ordinateurs et apprentissage*, Seymour PAPERT ; Flammarion - 1981

[PEY-88] « Un laboratoire pour l'enseignement de la programmation. » *Colloque sur l'évolution de l'outil Informatique à l'université*, Poitiers. J.-P. PEYRIN, J.-M. CAGNAT, V. GUÉRAUD, I. LIEM & S. PAINVIN - 1988

[RIC-81] *Initiation à l'algorithmique*, C. & P. RICHARD ; Belin - l981

[SCH-85] *Projet DIDALP ; DIDactitiel en ALgorithmique et Programmation,* P.C. Schol*l* ; Spécification du projet, rapport interne, LGI, IMAGrenoble - l985

[WIR-71] *Program Development by Stepwise Refinemen*t, Niklaus WIRTH ; CommACM, 14, 4, pages 221-227 - 1971

[WIR-76] *Algorithms+Data Structures=Programs,* N. WIRTH ; Prentice-Hall - 1976

[WIR-83] *Introduction à la programmation systématique,* N. WIRTH ; Masson - 1983